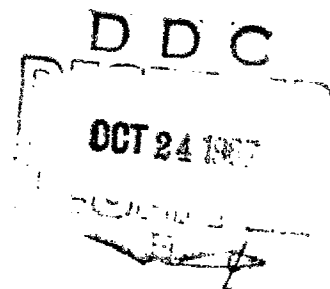EASY ENGLISH, A LANGUAGE FOR INFORMATION
RETRIEVAL THROUGH A REMOTE
TYPEWRITER CONSOLE

by

M. Rubinoff[†*]
S. Bergman*
H. Cautin*
F. Rapp*

April 1967

UNIVERSITY of PENNSYLVANIA

The Moore School of Electrical Engineering

PHILADELPHIA, PENNSYLVANIA 19104

Best
Available
Copy

University of Pennsylvania
THE MOORE SCHOOL OF ELECTRICAL ENGINEERING
Philadelphia, Pennsylvania

EASY ENGLISH, A LANGUAGE FOR INFORMATION
RETRIEVAL THROUGH A REMOTE
TYPEWRITER CONSOLE

by

M. Rubinoff[†][*]
S. Bergman[*]
H. Cautin[*]
F. Rapp[*]

April 1967

* The Moore School Information
Systems Laboratory
University of Pennsylvania

† Also with
Pennsylvania Research Associates, Inc.

University of Pennsylvania
THE MOORE SCHOOL OF ELECTRICAL ENGINEERING
Philadelphia, Pennsylvania

EASY ENGLISH, A LANGUAGE FOR INFORMATION
RETRIEVAL THROUGH A REMOTE
TYPEWRITER CONSOLE

Easy English is a natural command language designed to simplify
communication between man and machine through a remote typewriter console.
It is made up of readily recognized sentences of the English language, sen-
tences which any layman might be expected to use in everyday requests for
services or articles from a familiar source. Easy English has been developed
as a command language for retrieval of documents from a computerized data
base, specifically from the Moore School Information Systems Laboratory
(MSISL) files. It is intended for all information retrieval systems using
remote typewriter consoles in a conversational mode.

Easy English is imbedded in the MSISL retrieval program which provides
computer-directed search, computer-aided editing, and other forms of computer
assistance. The attached typewriter printout presents a typical man-machine
conversation which illustrates Easy English along with a number of features
of the Laboratory retrieval system. Note that the latter currently provides
the option of translation of the Easy English request into Symbolic Command
Language while searching the files; this is a convenience for those who
might like to learn Symbolic Language on their own and use its shorter but
more formal structure in place of Easy English.

Because Easy English is in fact real English, the only thing that the
searcher needs to learn is that requests for information from the system
should be formulated in the following syntactical form:

| Introductory Clause | | Document Clause | | Data Clause | .

The following sentences present five forms in which the same retrieval
request can be phrased in Easy English:

(1) PLEASE LOCATE EVERYTHING WRITTEN BY ROBERT PERKINS ABOUT EASIAC OR
    PSEUDO-COMPUTERS BETWEEN 1955 AND 1959 < >

(2) COULD YOU FIND FOR ME SOMETHING CONTAINED IN THE REPOSITORY CONCERNING
    EASIAC OR PSEUDO-COMPUTERS THAT WAS AUTHORED BY ROBERT PERKINS AFTER
    1954 AND BEFORE 1960 < >

(3) I NEED ALL THE AVAILABLE DOCUMENTS PUBLISHED DURING THE PERIOD 1955 TO
    1959 BY ROBERT PERKINS ON THE SUBJECTS OF EASIAC OR PSEUDO-COMPUTERS < >

(4) WE'RE INTERESTED IN HAVING REFERENCES AND MATERIAL ON EITHER PSEUDO-COMPUTERS OR EASIAC AUTHORED BY ROBERT PERKINS FROM 1955 TO 1959 < >

(5) I WOULD LIKE YOU TO HELP ME OBTAIN INFORMATION FROM YOUR LIBRARY RELATED TO EASIAC OR PSEUDO-COMPUTERS AND WRITTEN BY ROBERT PERKINS IN THE YEARS 1955 THROUGH 1959 < >

Notice that despite the differences in vocabulary, all of these statements follow the same basic pattern; for example,

| COULD YOU FIND FOR ME | | SOMETHING CONTAINED IN |

| THE REPOSITORY | | CONCERNING ... |

Typical examples of phrases acceptable in the three clause categories are:

Introductory clause

(1) I would like ...
(2) Please find for me ...
(3) I have need of ...
(4) I desire ...

Document clause

(1) ... documents in the system ...
(2) ... information ...
(3) ... any available book or article in the repository ...
(4) ... references from the files ...
(5) ... all the stuff ...

Data clause

(1) ... written by Carr between 1958 and 1965.
(2) ... published in 1960 on information retrieval and word association but not programming.
(3) ... dated September 1966 by J. H. Smith, Joe Doe but not K. L. Jones about analog computers.

In the event that a word appearing in either the introductory or the document clause is not recognized, the computer initiates a man-machine dialogue in order to determine whether the word is essential and, if so, to seek out a synonym in its vocabulary. Examples of such dialogues appear on the typewriter printout (Appendix A), and should be self-explanatory. Because the printout presents the Symbolic Command Language equivalent, the characteristics of the latter are described in the next section.

Symbolic Command Language

As mentioned above, Easy English was developed on the top of and imbedded in the Laboratory's existing information retrieval system using Symbolic Command Language. Each Easy English request is therefore transformed into an equivalent query expressed in the Symbolic Command Language, which may be printed out as in Appendix A or may be suppressed. The system interprets the Symbolic Command Language request, transforms it into a set of operations, and executes those operations, thereby producing a list of accession numbers for documents that satisfy the original Easy English request.

A retrieval request written in Symbolic Command Language consists of the word "RETRIEVE" followed by some specification of documents to be retrieved. The specification part of a retrieval request consists of index terms, delimiters, sector designators and logical operations. Parentheses may be used for grouping. The allowable operators are &, †, and +, corresponding to the English AND, AND NOT, and inclusive OR. The first two are of equal rank and higher than that of the third. Operators of equal rank are executed from left to right.

The following sector designators are used to indicate the context of the index terms [1]:

$A0	Date of indexing and indexer's code
$A1	Author
$A2	Date of issue
$A3	Title
$A4	Editors
$A9	Collection in which articles appear
$B	Index terms
$C	Added information

When the user has completed formulation of his retrieval request, in either Symbolic or Easy English language, the information system executes the request as illustrated in Appendix A. The user is told the number of documents found. The user then selects the types of information that he wishes to have printed for each document. The various types of information are: date of indexing and indexer's code, author, date of issue, title, issuers, page size, number of illustrations, number of pages, index terms (descriptors), and added information codes.

Operation of Easy English

As mentioned above, each Easy English retrieval request consists of three clauses. Each clause contains one or more words or phrases to which a syntax class value has been assigned. A word's value determines the clause in which it may appear.

The program sequentially extracts words or phrases from the user's request, locates the word in the dictionary, finds its assigned value, and stores this value on an introduction code list. The process continues until a word or phrase is found whose value indicates that this word or phrase belongs to the document clause. At this point, introduction transformations are applied to the introduction code list to test for a valid introductory clause. Finding such a clause, the program repeats the process for the document clause with the exception that a word whose value indicates that it belongs to the data clause is the signal that the document code list is complete. Words in the data clause which are listed in the dictionary have their code placed in a data list, whereas words not listed are treated as index terms and are themselves placed in the data list. When the data list is complete as signaled by a special end-of-message character, parentheses are put in the list to maintain the logical construction of the sentence. Also sector designators and operations are put in the list so as to be recognizable to the Symbolic Command Language. At this point, the system executes the command.

If the introductory or document clause is not valid, the user is requested to re-enter his message. (See Figure 1 for the macro flowchart of Easy English.)

Easy English Example

The following is an illustrative request in Easy English:

I WOULD LIKE YOU TO HELP ME OBTAIN INFORMATION RELATED
TO KASIAC OR PSEUDO-COMPUTERS AND WRITTEN BY PERKINS < >

Notice that the < > character is the special end-of message character mentioned above.

The following words are taken from the Introductory Clause Dictionary:

01	02	03	04	05
RETRIEVE	HAVING	HAVE NEED OF	IN NEED OF	FOR ME
FIND	KNOWING	LIKE	INTERESTED IN	FOR US
LOCATE	GETTING	NEED	LOOKING FOR	ME
OBTAIN	TRACKING DOWN	DESIRE		US

06	07	08	09	10
AM	CAN	I'D	HAVE	TO
ARE	COULD	WE'D	KNOW	
WAS	SHOULD	I'LL	SEE	
WERE	WOULD	WE'LL		

11	12	13	14	15
ALSO	COULD YOU	I'M	IN	I
AND	HELP ME	WE'RE	WITH	WE
AS WELL AS	PLEASE			
OR	YOU			

The following is taken from the document clause dictionary:

17

INFORMATION

The following are taken from the Data Clause Dictionary:

06	42	15	02
RELATED TO	OR	AND	WRITTEN BY

Introduction transformations include:

12 is deleted	07*03 → 03
10*01 → 02	14*02 → 02
15*03 → 03	11*11 → 11
10*09 → 02	02*05 → 02
01*05 → 01	15*07 → 08
08*03 → 03	03*11*03 → 03
13*04 → 03	02*11*02 → 02
06*04 → 03	01*11*01 → 01

The following are recognized as valid introductory clauses:

01, 03, 03*02

Also 17 is a valid document clause.

The introduction code list in our illustration becomes (written horizontally):

      15*07*03*12*10*12*01

which becomes

      08*03*10*01

      03*10*01

      03*02

which is a valid introductory clause.

The document clause is valid since 17 is the only code in the document code list.

The data clause is assembled into the following data list (written horizontally):

06	EASIAC	42	PSEUDO COMPUTERS	15	02	PERKINS	which becomes
06	(EASIAC	+	PSEUDO COMPUTERS)	15	02	PERKINS	which becomes
(06	(EASIAC	+	PSEUDO COMPUTERS)	&	(02	PERKINS))	which becomes
($B	(EASIAC	+	PSEUDO COMPUTERS)	&	($A1	PERKINS))	which becomes

RETRIEVE ($B (EASIAC + PSEUDO COMPUTERS) & ($A1 PERKINS))

This last string is sent to the system to be interpreted.

BIBLIOGRAPHY

1. Rubinoff, M. and White, J. F., Jr.; Establishment of the ACM Repository and the Principles of the IR System Applied to its Operation. Comm. ACM, 8:595, 1965.

2. Rubinoff, M.; Why Sigir? ACM SIGIR FORUM, Vol. IV, No. 2, 1967.

3. Rubinoff, M. and White, J. F., Jr.; Description of Cataloging and Indexing System for the ACM Repository. The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia. 1965.

4. Rubinoff, M., et al; The Moore School Information Systems Laboratory. The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia. April 1967.

5. Cautin, H. and Rapp, F.; Description of Easy English. The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia. April 1967. (internal report)

6. Lowe, Thomas C.; Design Principles for an On-Line Information Retrieval System. Ph.D. dissertation presented to The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia. 1966.

7. Fischer, Stephen B.; An Executive Control System for Information Retrieval via a Remote Console. M.S. thesis presented to The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia. 1966.

APPENDIX A

AN ACTUAL RETRIEVAL DEMONSTRATION

YOU MAY PROCEED.:=   PLEASE FIND FOR ME BOOKS
        CONCERNING STATISTICAL FUNCTIONS OR STANDARD
        DEVIATION, BUT NOT BUSINESS ORIENTED ZILKS
        INC---ENTITLED 'RUNCIBLE''I'.<>

{ Message is
{ entered.

PRINT? :=   YES<>

YOUR MESSAGE IS:

01] PLEASE FIND FOR ME BOOKS
02] CONCERNING STATISTICAL FUNCTIONS OR STANDARD
03] DEVIATION, BUT NOT BUSINESS ORIENTED ZILKS
04] ENTITLED 'RUNCIBLE''I'.

{ Edited message
{ is printed out
{ with lines
{ enumerated by
{ computer.

CORRECTIONS? :=   YES<>

LINE NO. :=   3<>
LINE(S):

      DEVIATION, BUT NOT BUSINESS ORIENTED<>

{ A line is
{ changed.

MORE? :=   YES<>

LINE NO. :=   0<>
LINE(S):

      DEAR COMPUTER:<>

{ A line is
{ added.

MORE? :=   NO<>
PRINT? :=   YES<>

YOUR MESSAGE IS:

01] DEAR COMPUTER:
02] PLEASE FIND FOR ME BOOKS
03] CONCERNING STATISTICAL FUNCTIONS OR STANDARD
04] DEVIATION, BUT NOT BUSINESS ORIENTED
05] ENTITLED 'RUNCIBLE''I'.

{ This is the
{ final request
{ in Easy English.

CORRECTIONS? :=   NO<>

WE DID NOT RECOGNIZE THE WORD DEAR   .IS THIS WORD ESSENTIAL TO
THE MEANING OF YOUR SENTENCE? :=   NO<>

A non-essential
word is challenged
and then ignored.

WE DID NOT RECOGNIZE THE WORD COMPUTER   .IS THIS WORD ESSENTIAL TO
   THE MEANING OF YOUR SENTENCE? := NO<>

```
RETRIEVE         (  $B      (       (STATISTICAL FUNCTIONS  ⌠Symbolic
        )      +   (STANDARD    DEVIATION        )       |Command
        )      †   (BUSINESS    ORIENTED         )   ..  |Language
        )      &   ( SA3  RUNCIBLE            I          ⌡Equivalent
        )
```

000001 'REFERENCES' HAVE BEEN RETRIEVED.

YOU MAY PROCEED.:=   PLEASE GET BOOKS BY--WRITTEN, EDITED ⌠Another
   OR PUBLISHED BY CARR.<> |request in
                                                           ⌡Easy English.

PRINT? := NO<>

```
RETRIEVE         (      ( SA1  CARR      + SA4  CARR       )
      +    ( SA5  CARR      )   )
```

000009 'REFERENCES' HAVE BEEN RETRIEVED.

PRINT SOME? := NO<>

YOU MAY PROCEED.:=   GET BOOKS BY EITHER CARR OR RUBINOFF ⌠A Third
   BUT NOT BY CARR<>   ⌡request.

PRINT? := NO<>

```
RETRIEVE          ( SA1      (      CARR      )   +   (
RUBINOFF          )    )    †    ( SA1 ,CARR
        )
```

00000: 'REFERENCES' HAVE BEEN RETRIEVED.

YOU MAY PROCEED.:=   OBTAIN FOR ME BOOKS WRITTEN IN 1961 <>
                                                          ⌠Request
                                                          ⌡number 4.

PRINT? := NO<>

RETRIEVE   SA2 1961

000127 'REFERENCES' HAVE BEEN RETRIEVED.

PRINT SOME? := NO<>

YOU MAY PROCEED.:=   I WOULD LIKE YOU TO FIND BOOKS ⌠Request number 5.
   WRITTEN, EDITED, AND PUBLISHED BY CARR.<> |Note the ability
                                                       |of the system to
                                                       |separate out the
PRINT? := NO<> |author, editor, and
                                                       ⌡publisher functions.
```
RETRIEVE         (     ( SA1  CARR      & SA4  CARR        )
      &    ( SA5  CARR      )   )
```

NO 'REFERENCES' HAVE BEEN RETRIEVED.

```
                    ( Start )
                        │
                        ▼
                  ┌───────────┐
                  │   0 → I   │
                  │   1 → J   │
                  └───────────┘
                        │
                        ▼
              ┌───────────────┐
              │   I + 1 → I   │────▶( δ )
              └───────────────┘
                        │
                        ▼
          ┌─────────────────────────┐
          │  Extract Ith word from  │────▶( γ )
          │  input sentence         │
          └─────────────────────────┘
                        │
                        ▼
          ┌─────────────────────────┐
          │  Locate word in         │────▶( ω )
          │  dictionary             │
          └─────────────────────────┘
                        │
                        ▼
                  ⬡ J : 3 ⬡────▶( α )
                        │
                        ▼
      ⬡ Word is      N     ⬡ Is word           Y    ⬡ Can user      N    ┌───────────┐
        in        ───────▶   essential to   ───────▶  supply a   ──────▶ │ Re-enter  │────▶(start)
        dictionary ⬡         the meaning of  ⬡        synonym  ⬡         │ message   │
                        │      sentence                   │               └───────────┘
                       Y│         N│                     Y│
                        ▼          ▼                      ▼
              ┌──────────────┐   ( δ )          ┌──────────────────┐
              │ Find its code│                  │ Obtain synonym   │────▶( ω )
              └──────────────┘                  └──────────────────┘
                        │
                        ▼
  ( π )◀─N  ⬡ Is code       ◀────  ⬡ J : 1 ⬡ ────▶  ⬡ Is code      N ──▶( ∅ )
            a valid                                    a valid
            introductory                               document
            clause ⬡                                   clause ⬡
                 │Y                                        │Y
                 │         ┌──────────────────┐            │
                 └───────▶ │ Place code at    │◀───────────┘
                           │ bottom of code   │
                           │ list             │
                           └──────────────────┘
                                    │
                                    ▼
                                  ( δ )
```
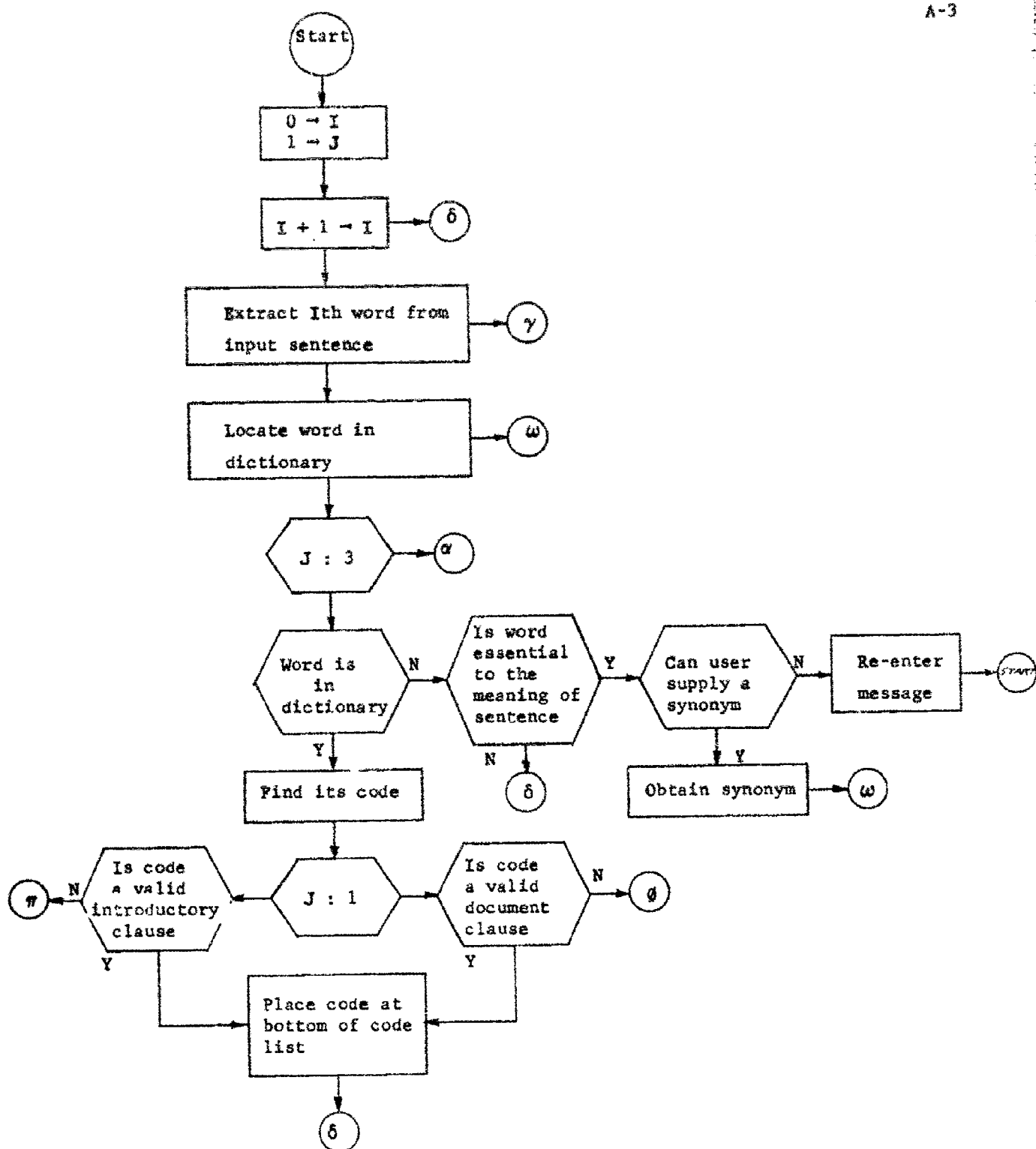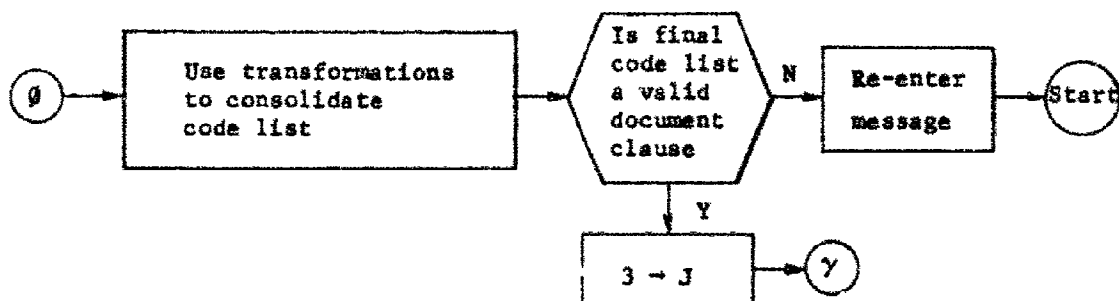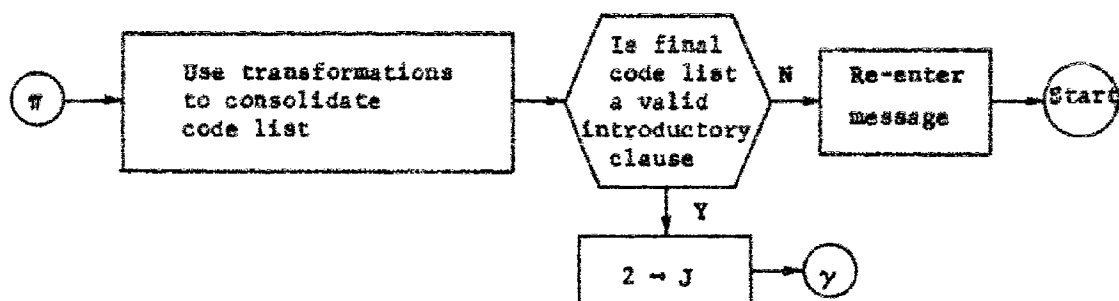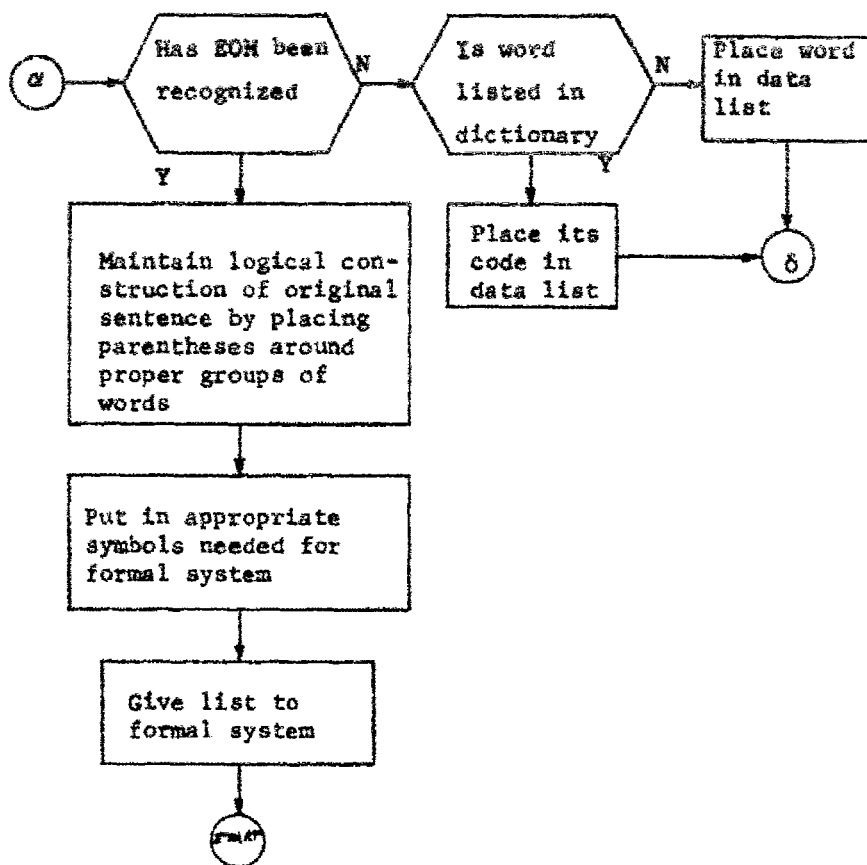
Figure 1-1

Figure 1-2

Figure 1-3